**Homework week 2**

**Basic data structures**

#include <bits/stdc++.h>

using namespace std;

int main()

{

int n;

cin >> n;

vector<int> a;

for (int i = 0; i < n; i++)

{

int x;

cin >> x;

a.push\_back(x);

}

sort(a.begin(), a.end());

int cnt = 0;

long long res = 0;

for (int i = 0; i < n; i++)

{

if (!i || a[i] != a[i - 1])

cnt = 1;

else

{

res += cnt;

cnt++;

}

}

cout << res;

return 0;

}

#include <bits/stdc++.h>

using namespace std;

struct Node

{

Node \*nextNode;

int value;

};

Node \*head = NULL;

void insertNode(int p, int x)

{

Node \*newNode = new Node();

newNode->value = x;

if (!p)

{

newNode->nextNode = head;

head = newNode;

return;

}

Node \*cur = head;

for (int i = 0; i < p - 1; i++)

{

cur = cur->nextNode;

}

newNode->nextNode = cur->nextNode;

cur->nextNode = newNode;

return;

}

void deleteNode(int p)

{

if (!p)

{

head = head->nextNode;

return;

}

Node \*cur = head;

for (int i = 0; i < p - 1; i++)

{

cur = cur->nextNode;

}

cur->nextNode = cur->nextNode->nextNode;

return;

}

int main()

{

int q;

cin >> q;

while(q--){

string d;

int p, x;

cin >> d;

if (d == "insert")

{

cin >> p >> x;

insertNode(p, x);

}

else

{

cin >> p;

deleteNode(p);

}

}

while (head != NULL)

{

cout << head->value << ' ';

head = head->nextNode;

}

return 0;

}

#include <bits/stdc++.h>

using namespace std;

struct Node

{

Node \*preNode, \*nextNode;

int value;

Node()

{

preNode = nextNode = NULL;

value = 0;

}

};

Node \*addNode(Node \*head, int x)

{

Node \*newNode = new Node();

newNode->value = x;

if (head == NULL)

{

head = newNode;

return head;

}

Node \*cur = head;

while (cur->nextNode != NULL)

{

cur = cur->nextNode;

}

newNode->preNode = cur;

cur->nextNode = newNode;

return head;

}

int count\_triplets(Node \*head)

{

int res = 0;

while (head != NULL)

{

if (head->preNode != NULL && head->nextNode != NULL)

{

if (head->preNode->value + head->value + head->nextNode->value == 0)

res++;

}

head = head->nextNode;

}

return res;

}

int main()

{

Node \*head = NULL;

int n;

cin >> n;

for (int i = 0; i < n; i++)

{

int x;

cin >> x;

head = addNode(head, x);

}

cout << count\_triplets(head);

return 0;

}

4.

#include <bits/stdc++.h>

using namespace std;

struct Node

{

Node \*nextNode;

int value;

};

Node \*head = NULL;

void enqueue(int x)

{

Node \*newNode = new Node();

newNode->value = x;

if (head == NULL)

{

head = newNode;

return;

}

Node \*cur = head;

while (cur->nextNode != NULL)

{

cur = cur->nextNode;

}

cur->nextNode = newNode;

return;

}

void dequeue()

{

head = head->nextNode;

return;

}

int main()

{

int q;

cin >> q;

while(q--){

string d;

int x;

cin >> d;

if (d == "enqueue")

{

cin >> x;

enqueue(x);

}

else

{

dequeue();

}

}

while (head != NULL)

{

cout << head->value << ' ';

head = head->nextNode;

}

return 0;

}

5.

#include <bits/stdc++.h>

using namespace std;

struct Node

{

Node \*nextNode;

int value;

};

Node \*head = NULL;

void push(int x)

{

Node \*newNode = new Node();

newNode->nextNode = head;

newNode->value = x;

head = newNode;

return;

}

void pop()

{

head = head->nextNode;

return;

}

int main()

{

int q;

cin >> q;

while (q--)

{

string d;

int x;

cin >> d;

if (d == "push")

{

cin >> x;

push(x);

}

else

{

pop();

}

}

while (head != NULL)

{

cout << head->value << ' ';

head = head->nextNode;

}

return 0;

}